ตอนที่ 11

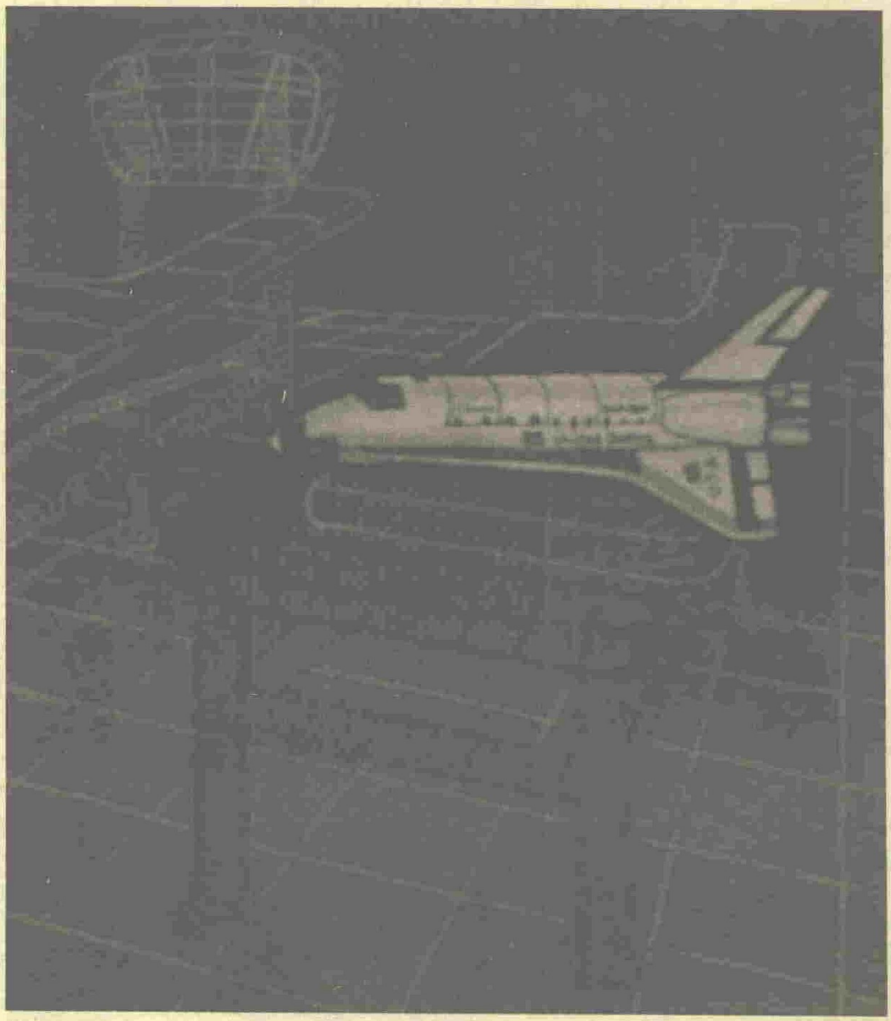
3D และ Shading

■ สุรพล พฤกษ์ไพบูลย์

ตามที่ได้เกริ่นเอาไว้แล้วว่าตอนนี้จะเป็นเรื่องของ 3D Animation หรือการแสดงความเคลื่อนไหวของรูปสามมิติในออโต้แคด อันจะเกี่ยวพันกับโปรแกรมอื่นอีกสองโปรแกรมคือ AutoShade และ AutoFlix ซึ่งโปรแกรมออโต้เชดเราจะนำมาใช้ในการลงสี (แสง) บนรูปสามมิติที่ได้จากการเขียนออโต้แคด และเราจะใช้ออโต้ฟลิกส์เมื่อเราต้องการแสดงภาพเคลื่อนไหว อันที่จริงในความหมายของแอนนิเมชันที่ถูกต้องจะเป็นการเคลื่อนไหวของวัตถุจริงบนจอภาพขณะนั้น มากกว่าการเตรียมภาพนิ่งขึ้นฉายต่อเนื่องกันแบบภาพยนตร์ อย่างวิธีที่ออโต้ฟลิกส์ใช้ อย่างไรก็ตามจะขอยกยอดไปกล่าวในตอนหน้าในเรื่องของการแสดงภาพเคลื่อนไหว สำหรับตอนนี้ก็จะขอนำเรื่องของ Shading มากล่าวกันก่อน เนื่องจากก่อนที่จะใช้โปรแกรมออโต้ฟลิกส์เพื่อแสดงภาพ เราควรจะได้ใช้และศึกษาเทคนิคบางประการในเรื่องของเชดดิ้ง (อันนี้ผมขอใช้ทับศัพท์นะครับ ถือว่าเป็นศัพท์เทคนิคก็แล้วกัน) โดยใช้โปรแกรมออโต้เชดกันเสียก่อน แต่ในตอนนี้จะขอไม่กล่าวลึกลงไปในเรื่องของเชดดิ้งมากนัก คือ เราจะไม่ไปสนใจในเรื่องของอัลกอริทึมหรือประวัติความเป็นมา ชนิดของการลงแสงเงา ฯลฯ อะไรทำนองนั้น หากแต่จะชี้ลงไปเฉพาะในส่วนที่จะนำไปใช้งาน และให้ต่อเนื่องไปถึงเรื่องของการแสดงภาพเคลื่อนไหวเท่านั้น ซึ่งในตอนนี้จะมีโปรแกรมสำหรับช่วยในการแก้ไขวัตถุที่จำเป็นสำหรับออโต้เชด

การใช้ออโต้เชด

ท่านที่เคยใช้โปรแกรมออโต้เชดมาแล้วขอให้อ่านข้ามย่อหน้านี้ไปได้เลยครับ เพราะผมจะขอกล่าวโดยย่อเกี่ยวกับการนำออโต้เชดมาใช้ร่วมกับออโต้แค้ด เพื่อให้ผู้ที่ไม่เคยใช้สามารถที่จะนำมาปะติดปะต่อเป็นเรื่องเป็นราวได้สะดวก ก่อนอื่นขอให้ท่านทราบเอาไว้ว่าโปรแกรมออโต้เชดเป็นโปรแกรมที่ขายแยกต่างหากจากออโต้แคด แต่จะมีโปรแกรมเวอร์ชันสำหรับใช้ทดลองมากับโปรแกรมออโต้แคด อันนั้นเราจะไม่ใช้นะครับ เพราะความสามารถค่อนข้างจะจำกัด ขอให้ใช้โปรแกรมที่ขายแยกมาต่างหาก ก่อนที่จะใช้ออโต้เชดเราจะต้องได้รูป ซึ่งเป็นไฟล์ที่มีสกุล .FLM จากออโต้แคดเสียก่อน ขอให้ท่านเขียนรูปสามมิติโดยใช้ 3DFACE หรือ Extruded entity คือวัตถุที่มีความหนา (Thickness) มากกว่า 0 (ขอให้เขียนรูปหลาย ๆ สี) ท่านจะสามารถเขียนรูปสามมิติได้ง่ายขึ้นมากถ้าท่านใช้โปรแกรม 3DUTIL ที่ผมได้ให้ไว้ในไมโครฯ ฉบับที่ 50 จากนั้นเราสามารถที่จะสร้างไฟล์สกุล .FLM ได้โดยใช้คำสั่ง FILMROLL การใช้คำสั่ง FILMROLL (หลังจากวาดรูปสามมิติแล้ว) นั้นกระทำได้สองวิธีคือ



1. เลือกมุมมอง 3 มิติ โดยสั่ง VPOINT เมื่อได้ภาพที่พอใจแล้วจึงใช้คำสั่ง FILMROLL เพื่อสร้างไฟล์ .FLM ตามต้องการ แต่เมื่อเข้าไปในออโต้เชดแล้วจะไม่มีแสงประดิษฐ์และมุมกล้องให้เลือก แสงที่ใช้ในการลงสีจะเกิดจาก Ambient light ซึ่งเป็นแสงที่เกิดจากความสว่างตามธรรมชาติในตอนกลางวันเท่านั้นเอง วิธีนี้เหมาะสำหรับการศึกษารูปทรงทางด้านสามมิติอย่างหยาบ ๆ เพื่อนำไปปรับปรุงรูปร่างแบบต่อไป

2. ในมุมมองปกติ คือรูปแปลน เราจะใช้โปรแกรมลิสป์ที่มากับออโต้เชดที่ชื่อว่า ASHADE.LSP เข้ามาช่วยเพื่อสร้างมุมมองและแสงต่าง ๆ ในการใช้งานปกติเราจะใช้วิธีนี้เสมอ เนื่องจากเราสามารถควบคุมแสงและความสว่างที่มีผลต่อรูปของเราได้ ซึ่งลักษณะของการนำไปใช้จะเป็นการใช้เพื่อแสดงงาน (presentation) เป็นส่วนใหญ่ การใช้งานก่อนที่จะเข้าไปในออโต้แคด จะต้องมีการเซตค่าที่เหมาะสมสำหรับสแต็กและฮีพดังนี้ คือ SET LISPSTACK = 5000 และ SET LISPHEAP = 40000 ซึ่งเป็นค่าที่จะใช้ในการเรียกและรันโปรแกรม ASHADE.LSP ภายในออโต้แคด

คำสั่งใน ASHADE. LSP

โปรแกรม ASHADE.LSP เมื่ออยู่ในหน่วยความจำแล้ว จะมีคำสั่งต่อไปนี้เพิ่มเติมคือ

LIGHT

เอาไว้สร้างแสงที่จะใช้ในรูป ซึ่งแสงนี้จะมีสองแบบคือ POINT เป็นแสงที่ออกมาจากแหล่งกำเนิดแสงทุกทิศทุกทาง ตัวอย่างของแสงชนิดนี้ก็คือ ดวงอาทิตย์ แสงชนิดที่สองก็คือ DIRECT เป็นแสงส่องแบบลำแสงขนานกัน เช่น แสงของสปอตไลต์ การใช้แสงเป็นสิ่งสำคัญเป็นอย่างมากในการแสดงงาน และมีสิ่งที่จะต้องคำนึงถึงหลายข้อที่จะใช้แสงให้ได้ผล

CAMERA

เป็นการสร้างกล้อง หรือมุมมอง เราจะต้องกำหนดตำแหน่งของผู้มอง ตามโคออร์ดิเนต X,Y,Z และตำแหน่งของเป้าหมายที่จะมองเป็น X,Y,Z ด้วยเช่นกัน ซึ่งโดยปกติออโต้แคดจะกำหนดมุมมองของสายตาปกติ คือเป็นมุมของกล้องที่ใช้เลนส์ 50 มม. ซึ่งจะมีประโยชน์ในการตรวจสอบว่าตามจุดที่ยืนนั้น จริง ๆ แล้วสามารถเห็นภาพได้อย่างไร อย่างไรก็ดี ในข้อนี้มีข้อโต้แย้งอยู่มากเกี่ยวกับภาพที่คนเรามองเห็นจริงกับภาพที่ออกมาจากกล้องถ่ายรูป เพราะคนเราสามารถรับภาพสามมิติในสมองก็ดี หรือแม้แต่เวลาเรามองภาพเราจะส่ายสายตาไปมาซ้ายขวาหรือขึ้นลง แล้วรับภาพทั้งหมดมาตีความรวมกัน มากกว่าจะมองเป็นจุดนิ่งแบบกล้องถ่ายรูป ฯลฯ ดังนั้นการสร้างภาพสามมิติตามมุมมองลักษณะนี้ เวลานำไปใช้จึงต้องระวังในข้อเหล่านี้เอาไว้สักหน่อยก็ดี จึงจะเป็นประโยชน์ได้

SCENE

เป็นการบันทึกข้อมูลเกี่ยวกับมุมมอง แสง และกล้องตามตำแหน่งต่าง ๆ เอาไว้เพื่อเวลานำมาลงสีในออโต้เชด ซึ่งคำสั่งซีนนี้จะทำหน้าที่ insert บล็อกสองชนิดคือ CLAPPER เป็นรูปเหมือนกับที่ใช้ในการถ่ายภาพยนตร์ คือเป็นป้ายบอกนั่นเอง และอีกชนิดหนึ่งคือ SHOT เป็นรูป / อยู่บนก้านของแคลปเปอร์ ช็อตขีดหนึ่ง ๆ จะเป็นชื่อกล้องหรือแสงที่ถูกรวมอยู่ในซีนนั้น ๆ จะเห็นได้จากถ้าเราต้องภารเคลื่อนย้ายแคลปเปอร์ เราจะต้องใช้วินโดว์โดยการกำหนด W หรือ C และจะมีจำนวน object หลายอัน

ความสมบูรณ์ที่มาพร้อมกับความบกพร่อง

เราจะเห็นว่าโปรแกรม ASHADE.LSP จะทำหน้าที่เป็นอินเตอร์เฟสระหว่างออโต้แคดและออโต้เชด ไม่มีข้อมูลทางเทคนิคใด ๆ จากออโต้แคดที่จะแสดงว่าคำสั่ง FILMROLL ทำอะไรกับบล็อกรูป กล้อง แสง และซีนเหล่านี้ นั่นคือ ASHADE.LSP ได้เตรียมอุปกรณ์ต่าง ๆ เอาไว้ให้เราอย่างสมบูรณ์ในการสร้างสรรค์โดยใช้แสงและมุมมองเพื่อให้เกิดรูปทรงแบบเปอร์สเปกตีฟ และความลึกทางสามมิติโดยใช้แสงเข้าช่วย ความสมบูรณ์แน่หรือ?... หาก...ปราศจากการแก้ไข!

SHADEATT.LSP

รูปของกล้อง แสง และซีน (แสดงเป็นรูปของแคลปเปอร์ที่ใช้ในการถ่ายหนัง) จะเป็นบล็อกพร้อมด้วยแอตทริบิวต์ และจะมีความสัมพันธ์กัน ไม่สามารถที่จะเคลื่อนย้ายหรือก็อบปี้โดยใช้คำสั่งออโต้แคดปกติ เพราะข้อมูลเกี่ยวกับตำแหน่งมองและเป้าหมายถูกบันทึกไว้ในแอตทริบิวต์ รวมทั้งการลบออกหรือแก้ไขก็ไม่สามารถจะทำได้โดยง่าย เช่น ถ้าเราลบกล้องหรือแสงออกโดยที่ไม่แก้ไขที่ซีนด้วยจะมีข้อผิดพลาด เพราะซีนจะอ้างถึงกล้องหรือแสงที่เราลบออกไปแล้วซึ่งไม่มีตัวตน วิธีที่ออโต้แคดแนะนำก็คือให้ลบออกแล้วสร้างขึ้นใหม่ ซึ่งทำให้เสียเวลามาก แต่เรามีวิธีที่ดีกว่านั้นคือ การใช้ออโต้ลิสป์เข้ามาช่วยเพื่อทำการแก้ไขสิ่งเหล่านี้ สิ่งที่ ASHADE.LSP ไม่ได้ทำเอาไว้แล้วเรามีหน้าที่ทำให้เต็ม ซึ่งผมได้รวบรวมโปรแกรมที่จะช่วยในงานเอาไว้ในไฟล์ที่ชื่อว่า SHADEATT.LSP นอกจากนี้แล้วจะต้องใช้ไฟล์ ACAD.LSP ตามที่ได้เคยให้ไว้ในไมโคร ฯ ฉบับที่ 47 หรือ 48

```
;Find data by a given key and entity name
(defun get_data (n e)
   (if (and (= (type n) 'INT) (= (type e) 'ENAME))
      (cdr (assoc n (entget e)))
   )
)
```

รูปที่ 1 ฟังก์ชัน get-date (ให้ใส่เอาไว้ใน ACAD.LSP)

```
;Replace data in the list
(defun replace_data (p q e)
   (if (assoc p e)
      (subst (cons p q) (assoc p e) e)
      (if (= p 8) (insertn (cons p q) 3 e) (insertn (cons p q) 4 e))
   )
)
```

รูปที่ 2 ฟังก์ชัน replace-data (ให้ใส่เอาไว้ใน ACAD.LSP)

(ท่านที่ไม่มีไมโครฯฉบับดังกล่าวสามารถสั่งซอฟต์แมล์ของฉบับนี้จะมี ACAD. LSP ที่รวม get-data และ replace-data ด้วย) และยังมีฟังก์ชันอีกสองอันคือ get-data และ replace-data ที่ได้ให้ไว้ในฉบับนี้ ขอให้บรรจุรวมเอาไว้ในไฟล์ ACAD.LSP อันที่เคยให้ไว้ในไมโครเล่มดังกล่าวด้วยครับ ที่ผมไม่ได้ให้ไว้ตอนนั้นเพราะเห็นว่าไม่เกี่ยวกับเนื้อความและยังไม่ได้ใช้ และสิ่งที่จะลืมไม่ได้ก็คือโปรแกรมที่ได้ให้ไว้ในฉบับนี้จะทำงานกับออโต้แคดเวอร์ชัน 10 ขึ้นไปเท่านั้นครับ

คำสั่งใน SHADEATT. LSP

เมื่อเรียกโปรแกรมเข้าไปในหน่วยความจำแล้ว โดยใช้ฟังก์ชัน (load "SHADEATT") ที่ช่อง command: แล้วจะมีคำสั่งเพิ่มเติมให้ต่อไปนี้ คือ

MV

เมื่อต้องการเคลื่อนย้ายตำแหน่งของวัตถุที่ได้จากโปรแกรม ASHADE. LSP คือ กล้องและแสง ซึ่งจะเป็นการเปลี่ยนตำแหน่งของผู้ยืนมอง แต่จุดเล็งหรือเป้าหมายยังคงที่ เพราะฉะนั้นเมื่อเปลี่ยนตำแหน่งแล้ว ตัวกล้องหรือแสงจะถูกหันเข้าหาเป้าหมายเดิม สำหรับซีนสามารถที่จะเคลื่อนย้ายโดยใช้คำสั่ง MOVE Window หรือ Crossing ได้ตามปกติ

TG

เมื่อต้องการเปลี่ยนตำแหน่งของเป้าหมาย (target point) โดยที่สามารถเลือกวัตถุคือ กล้องและแสงได้หลายตัวพร้อมกันโดยใช้ Window หรือ Crossing ทำให้ง่ายต่อการเปลี่ยนจุดเล็งหรือเป้าหมายของแสงหรือกล้องหลาย ๆ อันไปที่จุดเดียว สำหรับบล็อกรูปซีน แคลปเปอร์ และแสงแบบเป็นจุด หรือแสงแบบดวงอาทิตย์ ไม่จำเป็นต้องใช้คำสั่งนี้ เพราะไม่มีเป้าหมาย หรือถ้าถูกรวมเข้าไปในวัตถุที่เลือกด้วยก็ไม่จำเป็นจะต้อง remove ออก เพราะคำสั่งนี้จะมีการตรวจสอบเอาไว้แล้ว

DL

เมื่อต้องการลบกล้องหรือแสงออกจากรูป ซึ่งจะลบออกจากซีนที่มีกล้องหรือแสงนั้น ๆ ด้วย ส่วนแคลปเปอร์และซีนสามารถที่จะลบออกโดยใช้คำสั่ง ERASE Crossing หรือ Window ของออโต้แคดตามปกติ (ขอให้แน่ใจว่าวินโดว์ครอบคลุมทั้งรูปแคลปเปอร์ที่จะลบทั้งหมด)

เนื่องจากโปรแกรมที่ลงในฉบับนี้ค่อนข้างจะยาวและมีความซับซ้อนเกินกว่าจะอธิบายโดยละเอียด ดังนั้นท่านที่สนใจในการเขียนโปรแกรม ขอได้โปรดศึกษาจากคอมเมนต์ ซึ่งได้ให้ไว้มากพอสมควร ในที่นี้จะอธิบายหลักการกว้าง ๆ พอเป็นสังเขปเท่านั้น

การที่เราจะแก้ไขวัตถุที่ได้มาจาก ASHADE.LSP คือ กล้อง แสง และซีนได้นั้น เราจะต้องแก้ไขที่ตัวแอตทริบิวต์ที่มากับรูปด้วย ซึ่งแอตทริบิวต์ที่มากับวัตถุเหล่านี้ถูกเซตเอาไว้ให้มองไม่เห็น แต่เราจะสามารถดูหรือแก้ไขได้โดยใช้คำสั่ง DDATTE (ผมคิดว่ามาจากคำว่า Displayed Dialogue ATTribute Edit) จะเห็นว่ามีแอตทริบิวต์อยู่หลายตัว เช่น SNAME, GNAME, LAX, LAY และ LAZ ฯลฯ ขึ้นอยู่กับวัตถุคือ กล้องและแสง จะมีแอตทริบิวต์เป็น SNAME, GNAME, LAX, LAY, LAZ แคลปเปอร์ จะมี SNAME, CAMREF, LGTREF ช็อต จะมี OTYPE, ONAME, SNAME

SNAME

จะเป็นชื่อของกล้อง หรือแสง หรือซีน ขึ้นอยู่กับวัตถุ เป็นชื่อที่เราตั้งขึ้นเอง เช่น CAMERA1

GNAME

คือชื่ออีกชื่อหนึ่ง มีบล็อกรูปเดียวที่ใช้ชื่อนี้คือ แสงประดิษฐ์ประเภทสปอตไลต์

ชื่อของบล็อก คือ DIRECT ส่วน GNAME จะเป็น PARALLEL

LAX, LAY, LAZ

เป็นจุดเล็งหรือเป้าหมาย (target point) ของแสงหรือกล้อง ส่วนแสงประเภทจุดหรือแสงเหนือศีรษะ จะไม่มีเป้าคือ LAX, LAY, LAZ จะมีค่าเป็นศูนย์

CAMREF, LGTREF

จะเป็นสตริงที่มีชื่อของกล้องและแสงทั้งหมดที่อ้างถึงในซีนนั้น ๆ ในรูปของ Camera: <name> และ Lights: <name1>, <name2>....,<nameN> ตามลำดับ

OTYPE

เป็นชื่อของชนิดของวัตถุเช่น Camera และ Light เป็นต้น

ONAME

คือชื่อของกล้องหรือแสง ในที่นี้จะเป็นชื่อเดียวกับ SNAME

การทำงานโดยหลักใหญ่ ๆ จะเป็นการจัดการเกี่ยวกับ entity ที่เป็นบล็อก ซึ่งจะมีแอตทริบิวต์ตามหลังมาเป็น entity อีกเบอร์หนึ่ง (เบอร์ของ entity เราเรียกว่า ename) โดยมากผมจะเรียกบล็อกว่าเป็นส่วนหัว และแอตทริบิวต์ที่ตามมาเป็นส่วนหาง เราจะใช้ฟังก์ชันในออโต้ลิสป์คือ entnext เพื่อที่จะให้ได้ ename ของแอตทริบิวต์ แต่อย่าลืมว่าบล็อก ๆ หนึ่งอาจจะมีแอตทริบิวต์ตามมาหลายตัว ename สุดท้ายซึ่งเป็นขอบเขตสิ้นสุดของบล็อกหนึ่ง ๆ นั้นจะมีชื่อว่า SEQEND ในออโต้ลิสป์ไม่มีฟังก์ชันซึ่งใช้จัดการกับ entity ที่มีแอตทริบิวต์ติดมาได้สะดวก เราจึงต้องเขียนขึ้นเองเป็นฟังก์ชันประเภทอรรถประโยชน์ ซึ่งนอกจากใน SHADEATT.LSP แล้วยังสามารถที่จะนำไปใช้ที่อื่นที่จำเป็นได้อีก ฟังก์ชันที่ว่านี้คือ gather, egat และ get-val รูปแบบของการใช้งานและตัวอย่างของการใช้ฟังก์ชัน ขอให้ดูจากตัว definition เลยนะครับ ส่วนตัวของโปรแกรม MV TG และ DL เองนั้นคงจะอธิบายตัวมันเองได้ดี เพราะผมได้เขียนคอมเมนต์เอาไว้มากแล้ว โปรแกรมในฉบับนี้จะช่วยให้ท่านใช้ออโต้เชดได้สนุกและสะดวกขึ้นมาก เรียกว่าเป็นคู่หูของ ASHADE.LSP เลยทีเดียว ในตอนหน้าพบกันในเรื่องของ ออโต้เชดและออโต้ฟลิกส์กันต่อนะครับ สวัสดีครับ

รูปที่ 3 ไฟล์ SHADE.LSP

```
; SHADEATT.LSP    Utilities (MV,TG,DL) Move,Change,Delete ashade objects.
; Author:.....Surapon Preugpaibul
; Date:. ..... Jul 20,1989
, Last updated date:.. ..... Jul 31,1989
;
; Purpose:.....Update ashade entity attributes
;                which names mean nothing to general users.
; Command:
;                MV - move ashade enitiy; ie. camera.
;                TG - change target (aim) point of ashade enity.
;                DL - delete ashade objects: camera and lights
;
;---------------------------------------------------------------------
; ( egat <ename list> )  =  <attribute list>
;
; Returns  ( ( <ename> <Block name> ) ( <ename> <Att. tag> )....)
;
; To get the name of those returning from (gather ..)
;
(defun egat (l)
  (mapcar '(lambda (x)
    (list (get_data 2 x) x)
    ) l
  )
)
;---------------------------------------------------------------------
; ( get_val <attribute key name> <attribute list> )
; Return attribute value from attribute list
;
(defun get_val (s l) (get_data 1 (cadr(assoc s l))) )
;
;---------------------------------------------------------------------
; ( target.pt <list> )
;
; Return target points, searching from list of attributes.
;                                                              →
```

รูปที่ 3 (ต่อ)

```
  (setq q (targetpt l)          ; Target point
        bn (get_data 2 a)       ; Block name
        r (- (angle p q) d90)   ; New rotation angle
        r0 (get_data 50 a)      ; Old rotation angle
        r1 (rtod (- r r0))      ; Different in degree
  )
  ; don't rotate these objects
  (if (not (or
      (= "OVERHEAD" bn)
      (= "SHOT" bn)
      (= "CLAPPER" bn)
      ))
  ; In order to get quick response use ACAD command to update rotation angle
    (command "rotate" a "" p (strcat "<<" (rtos r1)))
  )
  (popvars)
  (command "undo" "end")
)
;
;-------------------------------------------------------------------------
; TG
;
; Change Target point of Ashade object(s), modify attributes.
; Works on several objects at a time.
; Discard old target points
;
(defun C:TG (/ a l n)
  (command "undo" "group")
  (prompt "Change Target point.\n")
  (pushvars '("blipmode" "highlight" "cmdecho"))
  (setqv '("blipmode" 0 "highlight" 1 "cmdecho" 0))
  (setq a (ssget) l (sslength a) n 0)
  (setvar "blipmode" 1)
  (setq q1 (getpoint "\nNew Target point: "))
  (while (< n l)
    (setq b (ssname a n) e (gather b) d (egat e))
    (setq i (get_data 10 b)          ; Insert point
          bn (get_data 2 b)          ; Block name
          r (- (angle i q1) d90)     ; New rotation angle
          r0 (get_data 50 b)         ; Old rotation angle
          r1 (rtod (- r r0))         ; Different in degree
    )
    (setvar "highlight" 0)
    ; don't do anything with these objects
    (if (not (or
        (= "OVERHEAD" bn)
        (= "SHOT" bn)
        (= "CLAPPER" bn)
        ))
      (progn
      (updateLA q1 d)                ; update target point in attribute
    ; In order to get quick response use ACAD command to update rotation angle
      (command "rotate" b "" i (strcat "<<" (rtos r1)))
      )
    )
    (setq n (1+ n))
  )
  (popvars)
  (command "undo" "end")
)
;-------------------------------------------------------------------------
; ( delshot <SNAME> )
```

รูปที่ 3 (ต่อ)

```
;
; Delete the blocks SHOT which have the same scene name.
;
(defun delshot (s / ss l n e f)
  (setq ss (ssget "X" (list(cons 2 "SHOT"))) ; get all blocks SHOT
        l (sslength ss) n 0)
  )
  ; Now find the shot that has the same scene name
  (while (< n l)
     (setq e (ssname ss n)                  ; Block head entity name
           f (entnext(entnext(entnext e))) ; attrib. SNAME
     )
     (if (= s (get_data 1 f))               ; found!
        (entdel e)                          ; then delete it
     )
     (setq n (1+ n))
  )
)
;-----------------------------------------------------------------
; DL
;
; Delete the object in the scene. Valid objects are CAMERA and LIGHT.
; Scene or Shot could be deleted by ERASE Crossing command.
;
; In order to delete the Ashade's object, we must modify block CLAPPER,
; therefore the object's name is not included.  And we must delete
; block SHOT (marked as / in the clapper) that has attribute tag OTYPE
; corresponding to the object we want to delete.
;
; NOTE: If object is CAMERA then the whole scene (CLAPPER) and shots
;       or lights ...will be deleted as well as the camera itself.
;
;       If UNDO is used after this DL command, the REGEN must be
;       issued in order to restore the original of the display of
;       attributes.
;

(defun C:DL ( / a b c otype oname e n l en scene camref cam scmark es ls
               lt ln li ol la fs sh shn shl xsh shs she shflag ip ipx)

  (command "undo" "group")
  (prompt "Delete an Ashade object.\n")
  (pushvars '("blipmode" "cmdecho"))
  (setqv '("cmdecho" 0 "blipmode" 0))
  (setq a (entsel) a (car a)      ; Object's entity name
        b (gather a)              ; Entity name list
        c (egat b)                ; Entity & block name list
        otype (caar c)              ; Block name or Object type in OTYPE tag
        oname (get_val "SNAME" c)     ; Object name in ONAME tag
        ; gets all BLOCKs named CLAPPER
        e (ssget "X" (list (cons 2 "CLAPPER")))
        n 0 l (sslength e)
  )
  (princ (strcat oname "\n"))
  (entdel a)
  (prompt "Now processing CLAPPER, Please wait...\n")
  (if (or(= otype "DIRECT")(= otype "OVERHEAD"))
    (setq otype "LIGHT" otag "LGTREF")
    (setq otag "CAMREF")                  ; otype is CAMERA
  )
  (while (< n l)
    (setq en (ssname e n)                 ; Block head
```

→

รูปที่ 3 (ต่อ)

```
        scene (entnext en)              ; Attribute SCENE
        camref (entnext scene)          ; Attribute CAMREF
    )
    (if (= otag "CAMREF")               ; the deleted object is CAMERA
      (progn
      (setq cam (get_data 1 camref)                 ; Camera: <name>
            cam (substr cam 9 (1+ (-(strlen cam)9)))  ; camera name
      )
      (if (= oname cam)
        (progn
        (setq scmark (get_data 1 scene))  ; scene name to be deleted
        (delshot scmark)          ; delete shots that are in the scene
        (entdel en)
      ))
      )
      (progn                            ; the deleted object is LIGHTS
      (setq es (entnext(entnext scene))  ; Attribute LGTREF
            ls (get_data 1 es)                  ; Lights: <lights str>
            ls (substr ls 9 (1+ (-(strlen ls)9))) ; lights names
            lt (strlen ls)                      ; length of string
            ln 1 ol "Light:"
      )
      (while (< ln lt)              ; Search for light name
        (setq li ln)
        (while (and (<= li lt) (/= " " (substr ls li 1)))
          (setq li (1+ li))
        )
        (setq la (substr ls ln (- li ln)) ln li)
        (if (= oname la)
          (setq scmark (get_data 1 scene)) ; Keep scene name
          (setq ol (strcat ol " " la)) ; keep light string name
        )
        (setq ln (1+ ln))
      )
      (setq fs (entget es)
            fs (replace_data 1 ol fs)   ; change lights name in attrib.
      )
      (entmod fs)                       ; change attrib. information
      (entupd en)                       ; update block head ename

      ; Now process block SHOT that has the same scene name
      (setq sh (ssget "X" (list(cons 2 "SHOT"))) shl (1- (sslength sh))
            shn shl shflag nil)

      ; Searching backward.  These entities are inserted that way.
      (while (and (>= shn 0) (null shflag))
        (setq xsh (ssname sh shn)               ; block head
              she (entnext(entnext(entnext xsh))) ; attribute SNAME
              shs (get_data 1 she)              ; scene name
        )
        (if (= shs scmark)              ; found!
          (progn
          (setq shotype (entnext xsh)     ; attrib. OTYPE
                shoname (entnext shotype) ; attrib. ONAME
          )
          (if (= oname (get_data 1 shoname))  ; shot is found!
            (setq shflag 't)                  ; toggle flag
          )
        ))
        (if (null shflag)(setq shn (1- shn)))
      )
      (if shflag
```

รูปที่ 3 (ต่อ)

```
(defun targetpt (l / m)
  (list (atof(get_val "LAX" l)) ; X coordinate
        (atof(get_val "LAY" l)) ; Y coordinate
        (atof(get_val "LAZ" l)) ; Z coordinate
  )
)
;
;-------------------------------------------------------------------
; ( updblk  <insert point> <rotation angle> <ename list> )
;
; Replace rotation angle and insert point of every entity belonging to blk.
;
(defun updblk (i a l / c)
  (princ "\nUdating related attributes, please wait...")
  (mapcar '(lambda (x)
    (setq c (entget (cadr x)))
    (if i (setq c (replace_data 10 i c)))
    (if a (setq c (replace_data 50 a c)))
    (entmod c)
    ) l
  )
  (entupd (cadar l))
)
;
;-------------------------------------------------------------------
; ( updateLA <target point> <ename list> )
;
; Update target point of block
;
(defun updateLA (p l / a)
  ; In order to work with 1st list (X Y Z), truncat 2nd list to be 3 elements
  (setq l (member (assoc "LAX" l) l))
  (mapcar '(lambda (x y)
       (if (= "LA" (substr (car y) 1 2))
         (progn
         (setq a (replace_data 1 (rtos x 2 6) (entget (cadr y))))
         (entmod a)
       ))
     ) p l
  )
)
;
;-------------------------------------------------------------------
; MV
;
; Move an ashade object and update rotation and insert point of
; all attributes that belong to the block.
;
(defun C:MV (/ a l e i p q r r0 r1 x)
  (command "undo" "group")
  (prompt "Move an Ashade object.\n")
  (pushvars '("blipmode" "highlight" "cmdecho"))
  (setqv '("blipmode" 0 "highlight" 1 "cmdecho" 0))
  (setq a (entsel) i (osnap (cadr a) "ins"))
  (setvar "blipmode" 1)
  (setq p (getpoint i "\nNew location: ")
        a (car a)           ; Head ename
        e (gather a)        ; Block & attribute ename list
        l (egat e)          ; Attribute information list
  )
  (setvar "highlight" 0)
  (command "move" a "" i p)
```

```
        (progn
        (setq ip (get_data 10 xsh)) ; keep the insertion point
        (entdel xsh)                ; delete SHOT block

        ; Now will move SHOT that has the same scene name to the LEFT
        (setq shn (1- shn))
        (while (>= shn 0)
          (setq xsh (ssname sh shn)                    ; block head
                she (entnext(entnext(entnext xsh)))    ; attribute SNAME
                shs (get_data 1 she)                   ; scene name
          )
          (if (= shs scmark)                  ; found!
            (progn
            (setq ipx (get_data 10 xsh))
            (command "move" xsh "" ipx ip)
            (setq ip ipx)
            )
          )
          (setq shn (1- shn))
        )
      ))
    )
    )
    (setq n (1+ n))
  )
  (popvars)
  (command "undo" "end")
)
; EOF SHADEATT.LSP
```

■

COPAM

EXCLUSIVE DISTRIBUTOR

บริษัท เอกปัญญา จำกัด

ตัวแทนจำหน่ายไมโครคอมพิวเตอร์ที่มีชื่อเสียงจากต่างประเทศ มีความประสงค์จะรับสมัครบุคคลซึ่งมีความรู้ ความสามารถ และรักความก้าวหน้าในการทำงาน ในตำแหน่งต่อไปนี้

1. เจ้าหน้าที่ฝ่ายการตลาด จำนวน 6 อัตรา มีคุณสมบัติดังนี้
   - วุฒิการศึกษาระดับ ปวช, ปวส จนถึงปริญญาตรี
   - ไม่จำกัดเพศชายหรือหญิง
   - อายุระหว่าง 20 ถึง 30 ปี
   - สามารถพูด อ่าน และเขียนภาษาอังกฤษดี
   - หากมีพาหนะส่วนตัว จะพิจารณาเป็นพิเศษ
2. เจ้าหน้าที่ฝ่ายบริการ จำนวน 6 อัตรา มีคุณสมบัติดังนี้
   - วุฒิการศึกษาระดับ ปวช, ปวส จนถึงปริญญาตรี (อิเล็กทรอนิคส์ สาขาคอมพิวเตอร์)
   - เพศชาย
   - อายุระหว่าง 20 ถึง 30 ปี
   - สามารถดู อ่าน และเขียนภาษาอังกฤษดี
   - หากมีพาหนะส่วนตัว จะพิจารณาเป็นพิเศษ

ผู้สนใจโปรดสมัครด้วยตนเอง หรือ ส่งใบสมัครพร้อมหลักฐานการศึกษา และรูปถ่าย มาภายในวันที่ 30 กันยายน 2532

บริษัท เอกปัญญา จำกัด
ห้อง 101 อาคารสินธร ถนนวิทยุ
ปทุมวัน กรุงเทพฯ 10330

ด่วน..!